

# Package ‘shortuuid’

February 27, 2026

**Title** Generate and Translate Standard UUIDs

**Version** 0.1.1

**Description** Generate and translate standard Universally Unique Identifiers (UUIDs) into shorter - or just different - formats and back. Also implements base58 encoders and decoders.

**License** MIT + file LICENSE

**Encoding** UTF-8

**RoxygenNote** 7.3.2

**LinkingTo** Rcpp

**Imports** Rcpp

**Suggests** testthat (>= 3.0.0)

**Config/testthat/edition** 3

**NeedsCompilation** yes

**Author** David Schoch [aut, cre] (ORCID:  
<<https://orcid.org/0000-0003-2952-4812>>)

**Maintainer** David Schoch <david@schochastics.net>

**Repository** CRAN

**Date/Publication** 2026-02-27 12:00:02 UTC

## Contents

base58_to_uuid . . . . .	2
bitcoin58_to_uuid . . . . .	2
flickr58_to_uuid . . . . .	3
generate_uuid . . . . .	3
is.base58 . . . . .	4
is.uuid . . . . .	4
uuid_to_base58 . . . . .	5
uuid_to_bitcoin58 . . . . .	5
uuid_to_flickr58 . . . . .	6
validate.uuid . . . . .	6
<b>Index</b>	<b>7</b>

base58\_to\_uuid      *Convert base58 to uuid*

---

**Description**

Convert base58 to uuid

**Usage**

```
base58_to_uuid(input, alphabet)
```

**Arguments**

input              character vector of base58 strings  
alphabet            character vector representing an alphabet

**Value**

character vector of uuids

---

bitcoin58\_to\_uuid      *Convert base58 bitcoin encoded character vector to uuid*

---

**Description**

Convert base58 bitcoin encoded character vector to uuid

**Usage**

```
bitcoin58_to_uuid(input)
```

**Arguments**

input              character vector of base58 strings

**Value**

character vector of uuids

---

flickr58_to_uuid	<i>Convert base58 flickr encoded character vector to uuid</i>
------------------	---

---

**Description**

Convert base58 flickr encoded character vector to uuid

**Usage**

```
flickr58_to_uuid(input)
```

**Arguments**

input	character vector of base58 strings
-------	------------------------------------

**Value**

character vector of uuids

---

generate_uuid	<i>Generate a random RFC4122 v4-compliant UUID</i>
---------------	--

---

**Description**

Generate a random RFC4122 v4-compliant UUID

**Usage**

```
generate_uuid(n = 1)
```

**Arguments**

n	number of ids to generate
---	---------------------------

**Value**

character vector of uuids

**Examples**

```
generate_uuid(n = 5)
```

`is.base58`*validate if character vector is base58 encoded*

---

**Description**

validate if character vector is base58 encoded

**Usage**

```
is.base58(x, alphabet)
```

**Arguments**

<code>x</code>	A character vector
<code>alphabet</code>	character vector representing an alphabet

**Value**

logical vector indicating if each element is a valid base58 string

---

`is.uuid`*check if object is of class uuid*

---

**Description**

check if object is of class uuid

**Usage**

```
is.uuid(x)
```

**Arguments**

<code>x</code>	A character vector
----------------	--------------------

**Value**

logical indicating if the input is a valid UUID

---

uuid_to_base58	<i>Convert uuid to base58</i>
----------------	-------------------------------

---

**Description**

Convert uuid to base58

**Usage**

```
uuid_to_base58(input, alphabet)
```

**Arguments**

input	character vector of uuids
alphabet	character vector representing an alphabet

**Value**

character vector of base58 encoded uuids

---

uuid_to_bitcoin58	<i>Convert uuid to base58 encoding of bitcoin</i>
-------------------	---

---

**Description**

Convert uuid to base58 encoding of bitcoin

**Usage**

```
uuid_to_bitcoin58(input)
```

**Arguments**

input	character vector of uuids
-------	---------------------------

**Value**

character vector of base58 encoded uuids

**Examples**

```
uuids <- generate_uuid(5)  
uuid_to_bitcoin58(uuids)
```

uuid\_to\_flickr58      *Convert uuid to base58 encoding of flickr*

---

**Description**

Convert uuid to base58 encoding of flickr

**Usage**

```
uuid_to_flickr58(input)
```

**Arguments**

input                  character vector of uuids

**Value**

character vector of base58 encoded uuids

**Examples**

```
uuids <- generate_uuid(5)
uuid_to_flickr58(uuids)
```

---

validate.uuid                  *validate if a string is a uuid*

---

**Description**

validate if a string is a uuid

**Usage**

```
validate.uuid(x)
```

**Arguments**

x                          A character vector

**Value**

logical indicating if the input is a valid UUID

# Index

base58\_to\_uuid, [2](#)  
bitcoin58\_to\_uuid, [2](#)  
  
flickr58\_to\_uuid, [3](#)  
  
generate\_uuid, [3](#)  
  
is.base58, [4](#)  
is.uuid, [4](#)  
  
uuid\_to\_base58, [5](#)  
uuid\_to\_bitcoin58, [5](#)  
uuid\_to\_flickr58, [6](#)  
  
validate.uuid, [6](#)